

APELLIDOS, Nombre:

Nº Matrícula:

**UPM FI Departamento de Lenguajes y Sistemas Informáticos e Ingeniería del Software.**  
*Examen de Programación II. 28 de Octubre de 2013. Primer Parcial.*

**Realización:** El test se realizará en la hoja de respuesta. Es **importante** que no olvidéis rellenar vuestros datos personales y el código clave de vuestro enunciado. Se pueden utilizar hojas aparte como borrador.

**Duración:** La duración total del test será de **50 minutos**.

**Puntuación:** El test se valora sobre **10 puntos**. Las preguntas tipo test pueden tener una única respuesta o varias respuestas, el enunciado lo deja claro. Cada pregunta con una única respuesta respondida correctamente vale 1 punto, e incorrectamente respondida resta 1/3 puntos. Si en una pregunta con una única respuesta se selecciona más de una respuesta, la pregunta se puntuará con 0 puntos. Para una pregunta con varias respuestas, cada afirmación correcta seleccionada suma  $1/n^{\circ}_{respuestas\_correctas}$  puntos, y cada afirmación incorrecta seleccionada resta  $1/n^{\circ}_{respuestas\_correctas}$  puntos. Las preguntas no contestadas suman 0 puntos en cualquier caso.

**Calificaciones:** Las calificaciones se publicarán en moodle como muy tarde el día **30 de Octubre de 2013**

**Revisión:** Las revisiones serán el día **4 de Noviembre de 2013** previa petición por correo electrónico.

### Primer Ejercicio

Sean los siguientes paquetes a y b definidos de la siguiente manera:

<pre>package a; import b.B;  public class A1 {     private void f() {         B b = new B();         b.g();         b.g(1);         A2 a = new A2();         a.f();     } }</pre>	<pre>package b;  public class B {     int g() {         return 0;     }      public int g(int a) {         return a + 1;     } }</pre>
<pre>package a; import b.B;  class A2 {     void f() {         B b = new B();         // .....     } }</pre>	

#### Pregunta 1

Señalar **todas** las afirmaciones verdaderas. Puede haber más de una afirmación **correcta**.

- A)** `g(int a)` es visible desde `f()` de `A1`, pero `g()` no lo es.
- B)** Ni `g(int a)` ni `g()` son visibles desde `f()` de `A1` porque `f` es privado.
- C)** `f()` de `A2` no es visible desde `f()` de `A1`.
- D)** `f()` de `A2` es visible desde `f()` de `A1`.

### Segundo Ejercicio

Según lo visto en clase sobre modularidad,

#### Pregunta 2

Señalar **todas** las afirmaciones verdaderas. Puede haber más de una afirmación **correcta**.

- A)** En Java, los paquetes son una forma de crear módulos.
- B)** En Java, un paquete no puede contener a otros paquetes.
- C)** En Java, dos clases pertenecientes a distintos paquetes se pueden llamar igual.
- D)** Un módulo que ofrece servicios se compone únicamente de una parte privada, llamada implementación.

**Tercer Ejercicio**

Dada la siguiente definición de clase:

```
public class ClaseX {
    private static int dato = 1;
    private int dato2;

    public ClaseX(int dato) {
        this.dato = dato;
        dato2++;
    }

    public String toString() {
        return dato + " " + dato2;
    }
}
```

**Pregunta 3**

Indicar cuál es la salida por consola del siguiente código.

```
public class Prueba2 {

    public static void main(String args[]) {
        ClaseX objeto1 = new ClaseX(5);
        ClaseX objeto2 = new ClaseX(3);
        System.out.println(objeto1 + " " + objeto2);
    }
}
```

Sólo una respuesta es correcta.

- A) 5 1 3 1
- B) 5 2 3 2
- C) 3 1 3 1**
- D) 3 2 3 2

**Cuarto Ejercicio**

Dado el siguiente código:

```
public class Ejercicio1 {

    public static void main(String args[]) {
        int[] miArray;
        for (int i = 0; i < miArray.length; i++) {
            miArray[i] = i;
        }
    }
}
```

**Pregunta 4**

Señalar todas las afirmaciones verdaderas. Puede haber más de una afirmación correcta.

- A) El código es correcto, compila y se puede ejecutar.
- B) El código compila y se ejecuta correctamente si declaramos `int[] miArray = null;`
- C) Falta inicializar la variable array `miArray`.**
- D) El código compila y se ejecuta correctamente si declaramos `int[] miArray = new int[5];`**

**Quinto Ejercicio**

Dada la siguiente clase que implementa el método buscar() y define un método main() para probarla:

```
public class Ejercicio2 {  
  
    public static int buscar(int[] array, int numero) {  
        for (int i = 0; i < array.length; i++) {  
            if (array[i] == numero) return i;  
        }  
        return -1;  
    }  
  
    public static void main(String args[]) {  
        int[] miArray = new int[5];  
        for (int i = 0; i < miArray.length; i++) {  
            miArray[i] = i;  
        }  
        System.out.print(buscar(miArray, 5));  
    }  
}
```

**Pregunta 5**

Indicar cuál será el resultado de ejecutar el main(). Sólo una respuesta es correcta.

- A) Sale un -1 por la consola**
- B) Es apropiado interrumpir el bucle for de búsqueda con un return.
- C) Sale un 5 por la consola
- D) No es una buena práctica de programación usar un return dentro de una construcción if

**Sexto Ejercicio**

Dada la siguiente clase que implementa el método buscar() y define un método main() para probarla:

```
public class Ejercicio3 {  
  
    public static boolean buscar(int[] array, int numero) {  
        boolean encontrado = false;  
        for (int i = 0; i < array.length; i++) {  
            if (array[i] == numero) encontrado = true;  
        }  
        return encontrado;  
    }  
  
    public static void main(String args[]) {  
        int[] miArray = new int[5];  
        for (int i = 0; i < miArray.length; i++) {  
            miArray[i] = i;  
        }  
        System.out.print(buscar(miArray, 5));  
        System.out.print(" ");  
        System.out.print(buscar(miArray, 4));  
    }  
}
```

**Pregunta 6**

Indicar cuál será el resultado de ejecutar el main(). Sólo una respuesta es correcta.

- A) false true**
- B) *El código no compila*
- C) false false
- D) true true

**Séptimo Ejercicio**

Dadas las siguientes clases:

<pre>public class Padre {     protected int p = 3;     protected int c = 3;      public Padre() {         c = p + 4;         p = c + 2;     }      void f() {         System.out.print(" padre_");     } }</pre>	<pre>public class Hijo extends Padre {     private int h;      public Hijo(int a) {         super();         h = p + a + 3;     }      void f() {         System.out.print(" hijo_ " + h);         super.f();     } }</pre>
--	---

Y la ejecución del siguiente *main()*:

```
public static void main(String[] args) {
    Hijo h = new Hijo(2);
    h.f();
}
```

**Pregunta 7**

Indicar cuál será el resultado de ejecutar el *main()*. Sólo una respuesta es correcta.

- A) padre hijo 14
- B) padre hijo 8
- C) hijo 8padre
- D) hijo 14padre**

**Octavo Ejercicio**

Cuál de las siguientes afirmaciones sobre el concepto sobrescribir (*override*) es cierta:

**Pregunta 8**

Sólo una respuesta es correcta.

- A) Sobrescribir se aplica cuando una clase define un método que existe en otra clase cualquiera con el mismo identificador (nombre), el tipo del valor retornado es el mismo y el tipo y número de los parámetros son los mismos
- B) Sobrescribir se aplica cuando una clase hija define un método que existe en la clase padre con el mismo identificador (nombre), el tipo del valor retornado es el mismo y el tipo y número de los parámetros son los mismos**
- C) Sobrescribir es cuando existen dos métodos con el mismo nombre y distinto número o tipo de parámetros en una clase
- D) Sobrescribir se aplica cuando se tienen dos atributos con el mismo nombre pero de distinto tipo dentro de una clase

**Noveno Ejercicio**

Cuál de las siguientes afirmaciones sobre el concepto sobrecarga (*overload*) es cierta:

**Pregunta 9**

Sólo una respuesta es correcta.

- A) Sobrecarga se aplica cuando se tienen dos atributos con el mismo nombre pero de distinto tipo dentro de una clase
- B) Sobrecarga se aplica cuando una clase define un método que existe en otra clase cualquiera con el mismo identificador (nombre) y el tipo y número de los parámetros son distintos
- C) Sobrecarga se aplica cuando una clase hija define un método que existe en la clase padre con el mismo identificador (nombre), el tipo del valor retornado es el mismo y el tipo y número de los parámetros son los mismos
- D) Sobrecarga es cuando existen dos métodos con el mismo nombre y distinto número o tipo de parámetros en una clase**

**Décimo Ejercicio**

Dadas las siguientes afirmaciones sobre los atributos de clase.

**Pregunta 10**

Indicar cuál de las siguientes afirmaciones es cierta (sólo una respuesta es correcta).

- A) Una clase con un atributo de clase tiene que tener obligatoriamente métodos de clase.
- B) Un atributo de clase siempre tiene que ser público.
- C) Un atributo de clase nunca puede ser static.
- D) Un atributo de clase tiene el mismo valor para todos los objetos de esa clase.**